SOFTWARE PROCESS IMPROVEMENT

COMP6100001 - SOFTWARE ENGINEERING
SESSION 05

UNIVERSITAS **BINA NUSANTARA**

SUBJECT MATTER EXPERT

Kenny Jingga, S.Kom., M.T.

SESSION 05 – SOFTWARE PROCESS IMPROVEMENT

LEARNING OBJECTIVES

At the end of this session, students will be expected to be able to:

- Understand the Concept and Importance of Software Process Improvement (SPI)
- Describe Key Models for Software Process Improvement
- Explore Strategies and Metrics for Effective Process Improvement
- Analyze Case Studies on Real-World SPI Implementation

SESSION 05 – SOFTWARE PROCESS IMPROVEMENT AND METRICS AND MEASUREMENTS

CONTENT OUTLINE

- Introduction to Software Process Improvement (SPI)
- Software Process Improvement Models
- Strategies for Enhancing Software Processes
- Measuring Process Improvement Success
- Case Studies and Real-world Examples

INTRODUCTION TO SOFTWARE PROCESS IMPROVEMENT (SPI)

OVERVIEW AND IMPORTANCE

- **Software Process Improvement (SPI)** refers to the practice of assessing and enhancing software development processes to improve quality, productivity, and predictability. SPI aims to identify weaknesses in current processes and implement changes that lead to better performance and outcomes.

Importance of SPI:

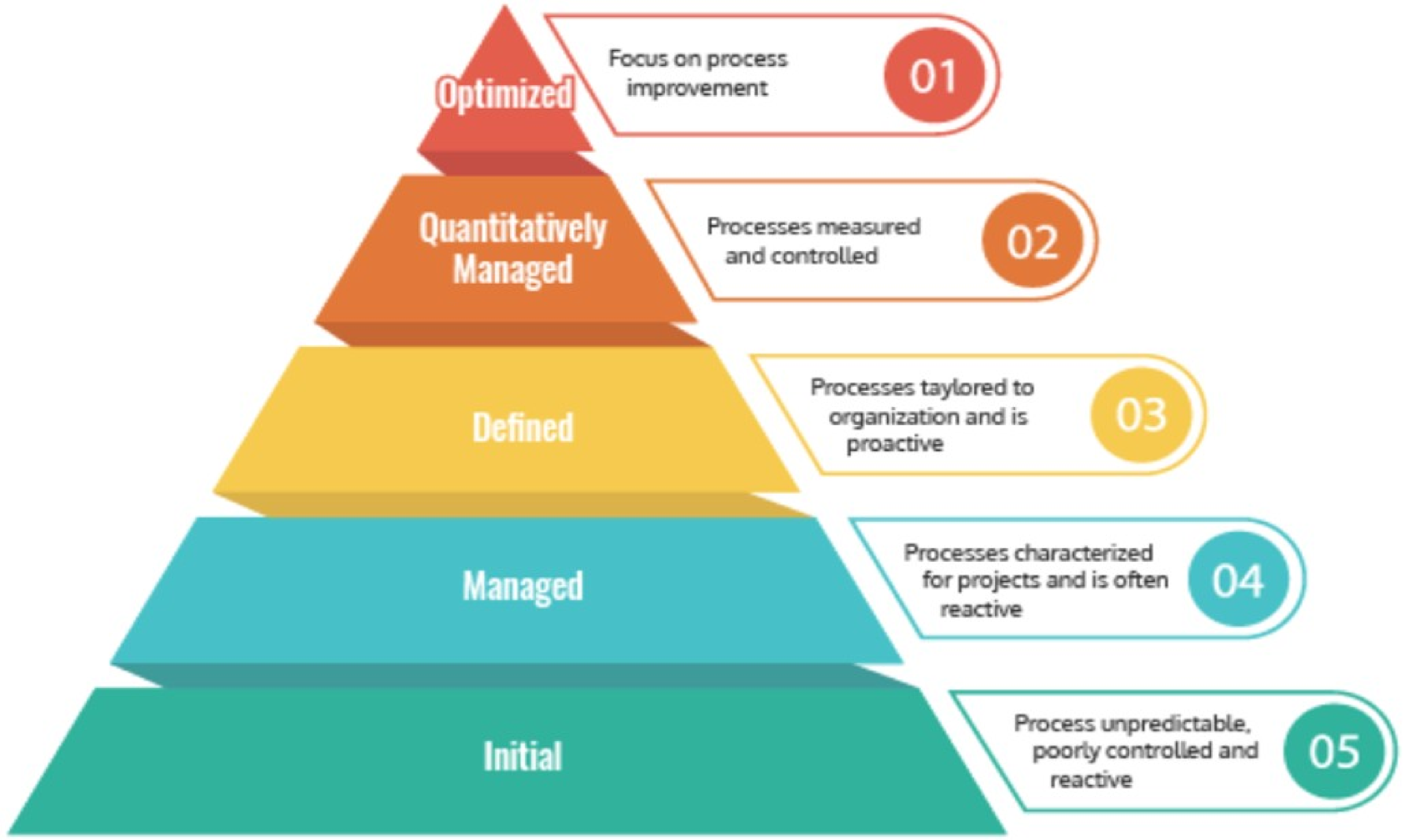
- **Quality Enhancement**: Improves the quality of software products.
- **Productivity Increase**: Streamlines processes to reduce time and cost.
- **Risk Reduction**: Identifies and mitigates potential issues early.
- **Customer Satisfaction**: Leads to higher satisfaction through reliable delivery.
- **Competitive Advantage**: Positions organizations better in the market.

BINUS UNIVERSITY

People
Innovation
Excellence

SOFTWARE PROCESS IMPROVEMENT MODELS

CAPABILITY MATURITY MODEL (CMM)


Optimized
Focus on process improvement
01
Quantitatively Managed
Processes measured and controlled
02
Defined
Processes taylored to organization and is proactive
03
Managed
Processes characterized for projects and is often reactive
04
Initial
Process unpredictable, poorly controlled and reactive
05

- **Definition**: The Capability Maturity Model (CMM) is a framework that describes the key elements of an effective software process.
- **Purpose**: Provides organizations with guidance on how to gain control of their processes for developing and maintaining software.

SOFTWARE PROCESS IMPROVEMENT MODELS

CAPABILITY MATURITY MODEL (CMM): MATURITY LEVELS

- **Initial (Level 1)**: Processes are unpredictable, poorly controlled, and reactive.
- **Repeatable (Level 2)**: Basic project management processes are established to track cost, schedule, and functionality.
- **Defined (Level 3)**: Processes are documented, standardized, and integrated into a standard process for the organization.
- **Managed (Level 4)**: Detailed measures of the software process and product quality are collected and controlled.
- **Optimizing (Level 5)**: Continuous process improvement is enabled by quantitative feedback and from piloting innovative ideas and technologies.

SOFTWARE PROCESS IMPROVEMENT MODELS

CAPABILITY MATURITY MODEL (CMM): KEY PROCESS AREA (KPAS)

- **Level 2 KPAs**: Requirements Management, Project Planning, Project Tracking and Oversight, Subcontract Management, Quality Assurance, Configuration Management.
- **Level 3 KPAs**: Organization Process Focus, Process Definition, Training Program, Integrated Software Management, Product Engineering, Intergroup Coordination, Peer Reviews.
- **Level 4 KPAs**: Quantitative Process Management, Software Quality Management.
- **Level 5 KPAs**: Defect Prevention, Technology Change Management, Process Change Management.

SOFTWARE PROCESS IMPROVEMENT MODELS

BENEFITS AND CRITICISMS OF CMM

- **Benefits:**
  - **Improved Quality**: Systematic processes lead to higher quality products.
  - **Predictability**: Better estimation of schedules and costs.
  - **Customer Satisfaction**: Enhanced reliability and delivery.
  - **Process Improvement**: Framework for continuous improvement.

- **Criticisms**:
  - **Cost and Time**: Implementation can be expensive and time-consuming.
  - **Rigidity**: May not suit all organizational cultures, especially smaller or more agile ones.

SOFTWARE PROCESS IMPROVEMENT MODELS

OTHER SPI MODELS: ISO/IEC 15504 (SPICE)

- **Definition**: An international standard for software process assessment.
- **Purpose**: Provides a framework for assessing the maturity of software development processes.
- **Process Capability Levels**: Incomplete, Performed, Managed, Established, Predictable, Optimizing.

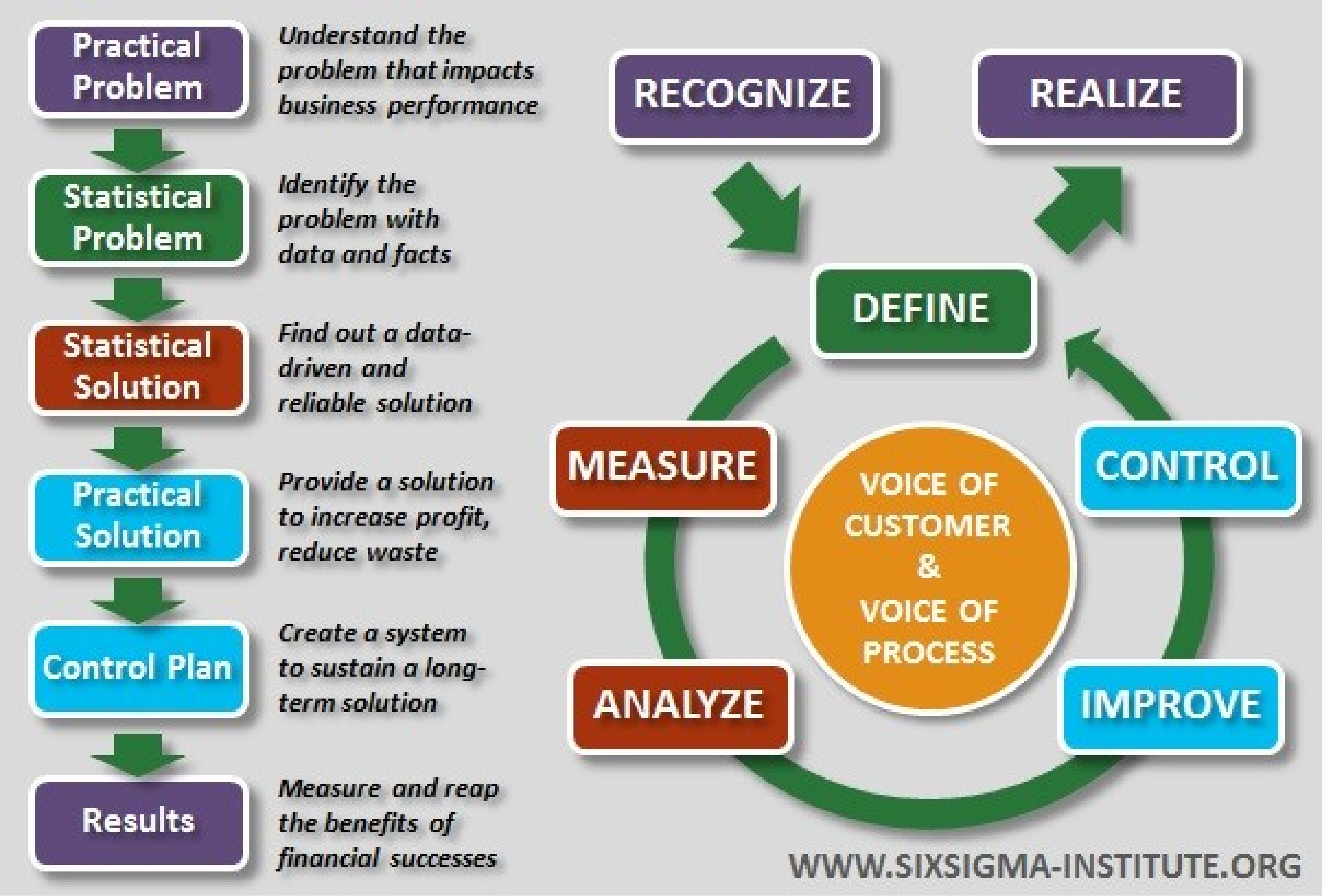
SOFTWARE PROCESS IMPROVEMENT MODELS

OTHER SPI MODELS: SIX SIGMA

- **Definition**: A data-driven approach for eliminating defects and improving quality in any process.

- **Purpose**: Aims for near perfection in process performance.

- **Key Concepts**: DMAIC methodology (Define, Measure, Analyze, Improve, Control).


Practical Problem
Understand the problem that impacts business performance
Statistical Problem
Identify the problem with data and facts
Statistical Solution
Find out a data-driven and reliable solution
Practical Solution
Provide a solution to increase profit, reduce waste
Control Plan
Create a system to sustain a long-term solution
Results
Measure and reap the benefits of financial successes
RECOGNIZE
REALIZE
DEFINE
MEASURE
CONTROL
VOICE OF CUSTOMER & VOICE OF PROCESS
ANALYZE
IMPROVE
WWW.SIXSIGMA-INSTITUTE.ORG

STRATEGIES FOR ENHANCING SOFTWARE PROCESSES

1. PROCESS ASSESSMENT

- **Purpose**:
  - Evaluate current processes to identify strengths and weaknesses.
  - Establish a baseline for improvement.
- **Methods**:
  - **Appraisals and Audits**: Formal evaluations against standards like CMM.
  - **Surveys and Interviews**: Gathering input from team members and stakeholders.

STRATEGIES FOR ENHANCING SOFTWARE PROCESSES

2. PROCESS MODELING AND ANALYSIS

- **Purpose**:
  - Understand and visualize existing processes.
  - Identify inefficiencies and bottlenecks.
- **Techniques**:
  - **Flowcharts**: Visual representations of process steps.
  - **Unified Modeling Language (UML)**: Standardized modeling language for specifying, visualizing, and documenting processes.

STRATEGIES FOR ENHANCING SOFTWARE PROCESSES

3. CHANGE MANAGEMENT

- **Importance**:
  - Effective SPI requires managing the human and organizational aspects of change.
  - Resistance to change can impede process improvement efforts.
- **Strategies**:
  - **Communication**: Clearly explain the need and benefits of changes.
  - **Training**: Equip team members with necessary skills and knowledge.
  - **Involvement**: Engage stakeholders in the change process.

STRATEGIES FOR ENHANCING SOFTWARE PROCESSES

4. CONTINUOUS IMPROVEMENT PRACTICES

- **Kaizen**:
  - **Definition**: A Japanese term meaning "change for better" or continuous improvement.
  - **Application**: Implement small, incremental changes regularly.
- **Plan-Do-Check-Act (PDCA) Cycle**:
  - **Plan**: Identify an opportunity and plan for change.
  - **Do**: Implement the change on a small scale.
  - **Check**: Analyze the results of the change.
  - **Act**: If successful, implement it on a larger scale.

MEASURING PROCESS IMPROVEMENT SUCCESS

KEY PERFORMANCE INDICATORS (KPIS)

- Quantifiable measures used to evaluate the success of an organization or process in meeting objectives.
- **Common Software Process KPIs**:
  - **Defect Density**: Number of defects per size unit (e.g., per 1,000 lines of code).
  - **Cycle Time**: Time taken to complete a process from start to finish.
  - **Customer Satisfaction Index**: Feedback scores from customers.
  - **Return on Investment (ROI)**: Financial benefits gained from process improvements.

MEASURING PROCESS IMPROVEMENT SUCCESS

METRICS AND MEASUREMENT TECHNIQUES

- **Importance of Metrics**:
  - Provides objective data to assess process performance.
  - Facilitates informed decision-making.
- **Types of Metrics**:
  - **Product Metrics:** Measure characteristics of the software product (e.g., size, complexity).
  - **Process Metrics**: Measure aspects of the development process (e.g., effort, time).
  - **Project Metrics**: Measure project progress and performance (e.g., schedule variance).

MEASURING PROCESS IMPROVEMENT SUCCESS

BENCHMARKING

- **Definition**:
  - Comparing one's processes and performance metrics to industry bests or best practices from other organizations.
- **Benefits**:
  - Identifies performance gaps.
  - Provides insights into best practices.
  - Sets realistic improvement goals.

CASE STUDIES AND REAL-WORLD EXAMPLES

CASE STUDY: IMPLEMENTING CMM IN A MID-SIZED SOFTWARE COMPANY

- **Background**:
  - A mid-sized software development company faced issues with project delays and high defect rates.
  - Decided to adopt CMM Level 3 to improve processes.

CASE STUDIES AND REAL-WORLD EXAMPLES

CASE STUDY: IMPLEMENTING CMM IN A MID-SIZED SOFTWARE COMPANY

- **Implementation Steps**:

1. **Assessment**: Conducted a gap analysis to identify areas needing improvement.
2. **Training**: Provided CMM training to staff.
3. **Process** Definition: Developed standardized processes and documentation.
4. **Pilot Projects**: Implemented new processes in select projects.
5. **Review and Refine**: Collected feedback and made necessary adjustments.

CASE STUDIES AND REAL-WORLD EXAMPLES

CASE STUDY: IMPLEMENTING CMM IN A MID-SIZED SOFTWARE COMPANY

- **Results**:
  - **Improved Quality:** Defect rates reduced by 30%.
  - **On-time Delivery:** Project delays decreased significantly.
  - **Customer Satisfaction:** Positive feedback from clients increased.

REFERENCES

- Paulk, M. C., Curtis, B., Chrissis, M. B., & Weber, C. V. (2017). Capability Maturity Model for Software, Version 2.0. Software Engineering Institute.
- Pressman, R. S. (2020). Software Engineering: A Practitioner's Approach (9th ed.). McGraw-Hill.